IN THE CLAIMS

1.   (Currently Amended)   A method for designing a computer program, comprising:

accessing a plurality of domain rules <u>for a military theory</u>, each domain rule being invariant;

displaying a plurality of business rules <u>for the military theory</u>, each business rule being variable;

selecting one or more business rules of the plurality of business rules in response to a user selection;

customizing the one or more business rules;

associating the one or more business rules with a procedure;

associating the domain rules with the procedure;

displaying a model representing the procedure; and

generating a code corresponding to the procedure in order to design a computer program.

2.   (Original) The method of Claim 1, further comprising:

collecting the domain rules and the business rules;

allocating the domain rules and the business rules to a plurality of use cases;

realizing the use cases; and

assessing the domain rules and the business rules in accordance with the realization.

3.   (Original) The method of Claim 1, further comprising:

checking a syntax of the code; and

providing a notification if a syntax error is detected.

4.   (Original) The method of Claim 1, further comprising:

checking a logical consistency of the code; and

providing a notification if a logical inconsistency is detected.

5.   (Original) The method of Claim 1, further comprising:

checking a compatibility between the model and the code; and

providing a notification if an inconsistency is detected.

6.   (Currently Amended) The method of Claim 1, wherein the model is expressed according to a ~~common~~ modeling language.

7.   (Currently Amended)   Logic for designing a computer program, the logic embodied in a <u>computer-readable</u> medium and <u>when executed by a computer</u> operable to:

access a plurality of domain rules <u>for a military theory</u>, each domain rule being invariant;

display a plurality of business rules <u>for the military theory</u>, each business rule being variable;

select one or more business rules of the plurality of business rules in response to a user selection;

customize the one or more business rules;

associate the one or more business rules with a procedure;

associate the domain rules with the procedure;

display a model representing the procedure; and

generate a code corresponding to the procedure in order to design a computer program.

8.   (Original) The logic of Claim 7, further operable to:

collect the domain rules and the business rules;

allocate the domain rules and the business rules to a plurality of use cases;

realize the use cases; and

assess the domain rules and the business rules in accordance with the realization.

9.   (Original) The logic of Claim 7, further operable to:

check a syntax of the code; and

provide a notification if a syntax error is detected.

10.   (Original) The logic of Claim 7, further operable to:

check a logical consistency of the code; and

provide a notification if a logical inconsistency is detected.

11.   (Original) The logic of Claim 7, further operable to:

check a compatibility between the model and the code; and

provide a notification if an inconsistency is detected.

12.   (Currently Amended) The logic of Claim 7, wherein the model is expressed according to a ~~common~~ modeling language.

13.   (Currently Amended)   A system for designing a computer program, comprising:

a database operable to store a plurality of domain rules <u>for a military theory</u>, each domain rule being invariant; and

a server coupled to the database and operable to:

display a plurality of business rules <u>for the military theory</u>, each business rule being variable;

select one or more business rules of the plurality of business rules in response to a user selection;

customize the one or more business rules;

associate the one or more business rules with a procedure;

associate the domain rules with the procedure;

display a model representing the procedure; and

generate a code corresponding to the procedure in order to design a computer program.

14.   (Original) The system of Claim 13, the server further operable to:

collect the domain rules and the business rules;

allocate the domain rules and the business rules to a plurality of use cases;

realize the use cases; and

assess the domain rules and the business rules in accordance with the realization.

15.   (Original) The system of Claim 13, the server further operable to:

check a syntax of the code; and

provide a notification if a syntax error is detected.

16.   (Original) The system of Claim 13, the server further operable to:

check a logical consistency of the code; and

provide a notification if a logical inconsistency is detected.

17.   (Original) The system of Claim 13, the server further operable to:

check a compatibility between the model and the code; and

provide a notification if an inconsistency is detected.

18.   (Currently Amended)   The system of Claim 13, wherein the model is
expressed according to a ~~common~~ modeling language.

19.   (Currently Amended)   A system for designing a computer program, comprising:

means for accessing a plurality of domain rules <u>for a military theory</u>, each domain rule being invariant;

means for displaying a plurality of business rules <u>for the military theory</u>, each business rule being variable;

means for selecting one or more business rules of the plurality of business rules in response to a user selection;

means for customizing the one or more business rules;

means for associating the one or more business rules with a procedure;

means for associating the domain rules with the procedure;

means for displaying a model representing the procedure; and

means for generating a code corresponding to the procedure in order to design a computer program.

20. (Currently Amended) A method for designing a computer program, comprising:

collecting a plurality of domain rules <u>for a military theory</u>, allocating the domain rules to a plurality of use cases, realizing the use cases, assessing the domain rules in accordance with the realization, and accessing the domain rules, each domain rule being invariant;

displaying a plurality of business rules <u>for the military theory</u>, each business rule being variable;

selecting one or more business rules of the plurality of business rules in response to a user selection;

customizing the one or more business rules;

associating the one or more business rules with a procedure;

associating the domain rules with the procedure;

displaying a model representing the procedure, the model expressed according to a ~~common~~ modeling language;

generating a code corresponding to the procedure in order to design a computer program;

checking a syntax of the code, and providing a notification if a syntax error is detected;

checking a logical consistency of the code, and providing a notification if a logical inconsistency is detected; and

checking a compatibility between the model and the code, and providing a notification if an inconsistency is detected.

21.   (Currently Amended) A method for managing rules for designing a computer program, comprising:

accessing a plurality of rules <u>for a military theory</u>;

analyzing the rules to separate a plurality of domain rules <u>of the military theory</u> from a plurality of business rules <u>of the military theory</u>, each domain rule being invariant, each business rule being variable;

storing the business rules; and

providing a business rule from the stored business rules in response to a request for the business rule.

22.   (Original) The method of Claim 21, further comprising:

customizing the provided business rule;

associating the customized business rule with a procedure; and

generating a code corresponding to the procedure in order to design a computer program.

23.   (Original) The method of Claim 21, further comprising:

associating the domain rules with a procedure; and

generating a code corresponding to the procedure in order to design a computer program.

24.   (Original) The method of Claim 21, further comprising:

allocating the domain rules and the business rules to a plurality of use cases;

realizing the use cases; and

assessing the domain rules and the business rules in accordance with the realization.

25.   (Currently Amended) A system for managing rules for designing a computer program, comprising:

a database operable to store a plurality of rules <u>for a military theory</u>; and

a server coupled to the database and operable to:

analyze the rules to separate a plurality of domain rules <u>of the military theory</u> from a plurality of business rules <u>of the military theory</u>, each domain rule being invariant, each business rule being variable;

store the business rules; and

provide a business rule from the stored business rules in response to a request for the business rule.

26.   (Original) The system of Claim 25, wherein the server is further operable to:

customize the provided business rule;

associate the customized business rule with a procedure; and

generate a code corresponding to the procedure in order to design a computer program.

27.   (Original) The system of Claim 25, wherein the server is further operable to:

associate the domain rules with a procedure; and

generate a code corresponding to the procedure in order to design a computer program.

28.   (Original) The system of Claim 25, wherein the server is further operable to:

allocate the domain rules and the business rules to a plurality of use cases;

realize the use cases; and

assess the domain rules and the business rules in accordance with the realization.

29.   (Currently Amended)   Logic for managing rules for designing a computer program, the logic embodied in a <u>computer-readable</u> medium and <u>when executed by a computer</u> operable to:

access a plurality of rules <u>for a military theory</u>;

analyze the rules to separate a plurality of domain rules <u>of the military theory</u> from a plurality of business rules <u>of the military theory</u>, each domain rule being invariant, each business rule being variable;

store the business rules; and

provide a business rule from the stored business rules in response to a request for the business rule.

30.   (Original) The logic of Claim 29, further operable to:

customize the provided business rule;

associate the customized business rule with a procedure; and

generate a code corresponding to the procedure in order to design a computer program.

31.   (Original) The logic of Claim 29, further operable to:

associate the domain rules with a procedure; and

generate a code corresponding to the procedure in order to design a computer program.

32.   (Original) The logic of Claim 29, further operable to:

allocate the domain rules and the business rules to a plurality of use cases;

realize the use cases; and

assess the domain rules and the business rules in accordance with the realization.

33. (Currently Amended) A system for managing rules for designing a computer program, comprising:

means for accessing a plurality of rules <u>for a military theory</u>;

means for analyzing the rules to separate a plurality of domain rules <u>of the military theory</u> from a plurality of business rules <u>of the military theory</u>, each domain rule being invariant, each business rule being variable;

means for storing the business rules; and

means for providing a business rule from the stored business rules in response to a request for the business rule.

34.   (Currently Amended) A method for managing rules for designing a computer program, comprising:

accessing a plurality of rules <u>for a military theory</u>;

analyzing the rules to separate a plurality of domain rules <u>of the military theory</u> from a plurality of business rules <u>of the military theory</u>, each domain rule being invariant, each business rule being variable;

allocating the domain rules and the business rules to a plurality of use cases;

realizing the use cases;

assessing the domain rules and the business rules in accordance with the realization;

storing the business rules;

providing a business rule from the stored business rules in response to a request for the business rule;

customizing the provided business rule;

associating the customized business rule with a procedure;

associating the domain rules with the procedure; and

generating a code corresponding to the procedure in order to design a computer program.

35.   (Currently Amended) A method for initiating display of a view of a computer program design, comprising:

accessing a plurality of artifacts of a computer program design, each artifact of the plurality of artifacts expressed using a ~~common~~ modeling language;

receiving a selection of a first view from a plurality of views, each view of the plurality of views associated with a display of a subset of the plurality of artifacts;

organizing a first subset of artifacts for display according to the first view, the first subset comprising a particular artifact;

initiating display of the first subset of artifacts according to the first view;

receiving a selection of a second view from the plurality of views;

organizing a second subset of artifacts for display according to the second view, the second subset comprising the particular artifact; and

initiating display of the second subset of artifacts according to the second view.

36.   (Previously Presented) The method of Claim 35, wherein an artifact of the plurality of artifacts comprises a requirement of the computer program design.

37.   (Previously Presented) The method of Claim 35, wherein:

receiving the selection of the first view further comprises receiving a selection of a high-level artifact view; and

organizing the first subset of artifacts further comprises organizing a plurality of high-level artifacts of the plurality of artifacts according to the high-level artifact view.

38.   (Previously Presented) The method of Claim 35, wherein:

the first view comprises a high-level artifact view, the high-level artifact view comprising a structural view; and

the second view comprises the structural view.

39.   (Previously Presented) The method of Claim 35, wherein:

the first view comprises a high-level artifact view, the high-level artifact view comprising a behavioral view; and

the second view comprises the behavioral view.

40.   (Previously Presented) The method of Claim 35, wherein:

the first view comprises a structural view, the structural view comprising an active class; and

the second view comprises a behavioral view, the behavioral view comprising the active class.

41.   (Currently Amended) A system for initiating display of a view of a computer program design, comprising:

a database operable to store a plurality of artifacts of a computer program design, each artifact of the plurality of artifacts expressed using a ~~common~~ modeling language; and

a server coupled to the database and operable to:

receive a selection of a first view from a plurality of views, each view of the plurality of views associated with a display of a subset of the plurality of artifacts;

organize a first subset of artifacts for display according to the first view, the first subset comprising a particular artifact;

initiate display of the first subset of artifacts according to the first view;

receive a selection of a second view from the plurality of views;

organize a second subset of artifacts for display according to the second view, the second subset comprising the particular artifact; and

initiate display of the second subset of artifacts according to the second view.

42.   (Previously Presented) The system of Claim 41, wherein an artifact of the plurality of artifacts comprises a requirement of the computer program design.

43.   (Previously Presented) The system of Claim 41, wherein the server is further operable to:

receive the selection of the first view by receiving a selection of a high-level artifact view; and

organize the first subset of artifacts by organizing a plurality of high-level artifacts of the plurality of artifacts according to the high-level artifact view.

44.   (Previously Presented) The system of Claim 41, wherein:

the first view comprises a high-level artifact view, the high-level artifact view comprising a structural view; and

the second view comprises the structural view.

45. (Previously Presented) The system of Claim 41, wherein:

the first view comprises a high-level artifact view, the high-level artifact view comprising a behavioral view; and

the second view comprises the behavioral view.

46. (Previously Presented) The system of Claim 41, wherein:

the first view comprises a structural view, the structural view comprising an active class; and

the second view comprises a behavioral view, the behavioral view comprising the active class.

47.   (Currently Amended) Logic for initiating display of a view of a computer program design, the logic embodied in a <u>computer-readable</u> medium and <u>when executed by a computer</u> operable to:

access a plurality of artifacts of a computer program design, each artifact of the plurality of artifacts expressed using a ~~common~~ modeling language;

receive a selection of a first view from a plurality of views, each view of the plurality of views associated with a display of a subset of the plurality of artifacts;

organize a first subset of artifacts for display according to the first view, the first subset comprising a particular artifact;

initiate display of the first subset of artifacts according to the first view;

receive a selection of a second view from the plurality of views;

organize a second subset of artifacts for display according to the second view, the second subset comprising the particular artifact; and

initiate display of the second subset of artifacts according to the second view.

48.   (Previously Presented) The logic of Claim 47, wherein an artifact of the plurality of artifacts comprises a requirement of the computer program design.

49.   (Previously Presented) The logic of Claim 47, further operable to:

receive the selection of the first view by receiving a selection of a high-level artifact view; and

organize the first subset of artifacts by organizing a plurality of high-level artifacts of the plurality of artifacts according to the high-level artifact view.

50.   (Previously Presented) The logic of Claim 47, wherein:

the first view comprises a high-level artifact view, the high-level artifact view comprising a structural view; and

the second view comprises the structural view.

51.   (Previously Presented) The logic of Claim 47, wherein:

the first view comprises a high-level artifact view, the high-level artifact view comprising a behavioral view; and

the second view comprises the behavioral view.

52.   (Previously Presented) The logic of Claim 47, wherein:

the first view comprises a structural view, the structural view comprising an active class; and

the second view comprises a behavioral view, the behavioral view comprising the active class.

53.   (Currently Amended) A system for initiating display of a view of a computer program design, comprising:

    means for accessing a plurality of artifacts of a computer program design, each artifact of the plurality of artifacts expressed using a ~~common~~ modeling language;

    means for receiving a selection of a first view from a plurality of views, each view of the plurality of views associated with a display of a subset of the plurality of artifacts;

    means for organizing a first subset of artifacts for display according to the first view, the first subset comprising a particular artifact;

    means for initiating display of the first subset of artifacts according to the first view;

    means for receiving a selection of a second view from the plurality of views;

    means for organizing a second subset of artifacts for display according to the second view, the second subset comprising the particular artifact; and

    means for initiating display of the second subset of artifacts according to the second view.

54.   (Currently Amended) A method for initiating display of a view of a computer program design, comprising:

accessing a plurality of artifacts of a computer program design, each artifact of the plurality of artifacts expressed using a ~~common~~ modeling language, an artifact of the plurality of artifacts comprising a requirement of the computer program design;

receiving a selection of a first view from a plurality of views, each view of the plurality of views associated with a display of a subset of the plurality of artifacts, the first view comprising a high-level artifact view, the high-level artifact view comprising a structural view and a behavioral view;

organizing a first subset of artifacts for display according to the first view, the first subset comprising a particular artifact;

initiating display of the first subset of artifacts according to the first view;

receiving a selection of a second view from the plurality of views, the second view comprising at least one of the structural view and the behavioral view;

organizing a second subset of artifacts for display according to the second view, the second subset comprising the particular artifact; and

initiating display of the second subset of artifacts according to the second view.